OK DTIC

1

Knowledge Systems Laboratory Report KSL-85-9

A0 4649

[redacted]

85-0161

Final Report

Heuristic Programming Project

October 1979 -- September 1982

Principal Investigators

Professor Edward Feigenbaum
Professor Bruce Buchanan

Heuristic Programming Project
Stanford University
701 Welch Road, Bldg. C
Palo Alto, Ca. 94304

27 March 1985

Approved for Public Release: Distribution Unlimited

Prepared for

DARPA/IPTO
1400 Wilson Boulevard
Arlington, Va. 22209

[redacted]

Table of Contents

Final Report

Heuristic Programming Project

October 1979 -- September 1982

1. Introduction

This report will summarize the research work that was performed at Stanford University under ARPA Order No. 3423, Contract MDA903-80-C-107. The overall purpose of the contract was to support basic and applied research into the science and engineering of knowledge-based systems. This work had four major subcomponents:

1) Basic Research in Knowledge Acquisition, Representation, Utilization, and Evaluation,
2. Development of Tools to Facilitate the Process of Knowledge-Based Systems Construction,
3. Propagation of the Tools and Techniques of this Discipline to Other Areas of DOD Interest,
4. Initial Work on the Application of Knowledge Engineering Methodologies to Problems in VLSI Design.

All of the research in the first three subcomponents and a portion of the VLSI Design work were performed directly within the Heuristic Programming Project (HPP), Principal Investigators Professors Edward Feigenbaum and Bruce Buchanan. Associated with the VLSI Design work was a separate project in VLSI Theory, Principal Investigator Professor Jeffrey Ullman. Most of the research was performed using the facilities of the NIH-supported SUMEX-AIM computer system (a Digital Equipment Corporation dual KI-10 processor system). This contract did fund the acquisition of a DEC VAX 11/780 facility which supported some of the basic research into heuristic methods as well as the VLSI Theory work.

The remainder of this report will discuss both ~~the~~ HPP research and the VLSI Theory research, along with resulting publications.

2. HPP Research

2.1. Introduction

The HPP is a group of professors, research scientists, programmers, and students within the Stanford University Computer Science Department. It has long-standing research goals of applying the methodologies of applied artificial intelligence to difficult problems in science, engineering, and medicine. To accomplish these goals, HPP work ranges from basic research in the fundamentals of knowledge representation and acquisition, to specific projects to produce functional problem-solving system in areas as diverse as infectious disease treatment and design of integrated circuits.

In the context of this ARPA contract, HPP work was funded in the following seven areas:

1. Research on the Representation, Acquisition, and Use of Knowledge,
2. Generalization of Knowledge-Based System Design and Implementation Techniques (The AGE Project),
3. Research in Experiment Planning (The MOLGEN Project),

4. Development of Knowledge-Based 3-D Signal Understanding Techniques (The CRYSALIS Project)

5. Knowledge-Based VLSI Design

6. Creation of the *Handbook of Artificial Intelligence*

7. Transfer of AI Technology to Other Areas of DOD Interest

For each of these areas, the work that resulted during the course of the contract will be described along with pointers to the relevant HPP publications. A complete bibliography of HPP publications for the period of the contract is given at the end of this section.

2.2. Research on the Representation, Acquisition, and Use of Knowledge

In this period, our basic research on the representation, use and acquisition of knowledge focused on several large systems, described below. Numerous publications address these issues in the context of these systems. In addition, these issues are also central to other work on AGE, Experiment Planning, and VLSI Design discussed elsewhere. These have been, and continue to be, central issues in all of artificial intelligence.

The systems in which this research was accomplished are important in their own right. One of the fundamental conclusions of this work is that construction of a large, intelligent system depends on the separation of domain-specific knowledge from the "inference engine" that uses that knowledge. The systems listed below are demonstrations of the power of this conclusion. Publications listed with them discuss other conclusions and problems.

2.2.1. The Family of EMYCIN Systems (EMYCIN, ROGET, PUFF)

The research on EMYCIN, and subsequent dissemination of EMYCIN to many academic, industrial, and military sites, demonstrates the power of using a simple, rule-based representation of knowledge and a simple, backward-chaining inference mechanism. It was the first system to achieve the clear separation of inference procedures from knowledge base, and the first to define a domain-independent "shell" into which knowledge of many different domains could be encoded. Simply put, EMYCIN resulted from taking the powerful rule-based structure of MYCIN, along with associated explanation and knowledge acquisition facilities, but removing all domain-specific knowledge (infectious disease diagnosis and therapy for MYCIN). The EMYCIN work was co-funded by NSF Grant MCS79-03753. For further information, see HPP-80-11, HPP-80-22, HPP-81-16, and HPP-81-27.

PUFF was one of the first demonstrations that the EMYCIN framework could be used in domains other than the domain of the original MYCIN system. It showed the applicability of EMYCIN to diagnostic problems generally, and also showed a mechanism for technology transfer of AI ideas from the development environment to the environment of routine use. PUFF diagnosed lung diseases of various types and was successfully tested in a clinical setting at the Pacific Medical Center by Dr. Robert Fallat. It was co-funded by the NIH Institute of Medical Sciences, Grant 2522-2-02. For further information, see HPP-82-13.

ROGET was an experimental system whose focus is knowledge engineering, the method of acquiring knowledge interactively through a dialogue between a subject expert and a software specialist (called a knowledge engineer). ROGET encodes knowledge about the knowledge engineering process in order to carry on a dialogue itself with a subject expert. While previous work (e.g., TEIRESIAS) had focused on semi-automatic refinement of a knowledge base already designed and largely written, the research on ROGET focused on acquiring the initial conceptual structure needed to design the knowedge base in the first place. ROGET's special expertise was in the construction of systems, like PUFF, that are constructed in EMYCIN. It demonstrated that some parts of this knowledge engineering process can be automated, and also highlighted many parts of it that are still more an art than a science. For further information, see HPP-83-24.

2.2.2. Variations on the EMYCIN Architecture (CENTAUR, VM)

Two systems were developed in this time period that are variations on the simple representation and inference mechanism used in EMYCIN. CENTAUR addresses the problem of encoding strategy (or control) knowlege in a rule-based system and the issue of the expressive power of rule-based representations. It uses a combination of rules and frames to separate (a) the knowledge that makes inferences from data from (b) the knowledge that controls the focus of attention of the whole system. For further information, see HPP-80-17.

VM addresses the problem of making inferences that relate events in time. While EMYCIN implicitly assumes that the data constitute a "snapshot" of a situation at a moment in time, VM is written to make successive inferences using data that are continually arriving over time. VM was successfully tested in the realtime environment of a surgical recovery intensive care unit. It was co-funded by NIH Institute of Medical Sciences Grant 2522-2-02. For further information, see HPP-80-31.

2.2.3. EURISKO, GLISP, MRS

These are three additional domain-independent reasoning systems on which considerable research was performed. EURISKO uses a frame-based representation of starting assumptions and definitions in order to discover new knowledge in a domain through opportunistic search of the interesting combinations of the primitive concepts. It is a domain-independent outgrowth of the AM (Automated Mathematician) work of Professor Douglas Lenat. EURISKO was tested on domains as varied as war gaming and integrated circuit design architectures. It was co-funded by ONR Contract N00014-80-C-0609. For further information, see HPP-80-26 HPP-81-22, HPP-82-25, and HPP-82-26.

GLISP is a large package built on InterLisp that provides a conversational environment in which object-oriented systems can be constructed. It manages context and provides a graphical interface that frees the knowledge system programmer from many low-level programming concerns. GLISP was built by Professor Gordon Novak of the University of Texas while a visiting faculty member at Stanford. For further information, see HPP-82-1.

MRS is a framework for building reasoning systems within a logic-based representation. It provides a toolkit for predicate calculus systems that was previously available only for rule and frame-based systems. Included with MRS are a diverse repertory of commands for asserting and retrieving information, with various representations, inference techniques, and search strategies. What differentiates MRS from other knowledge representation systems is its ability to observe, reason about, and control its own activity. Within MRS, the system is treated as a domain in its own right. MRS work was co-funded by ONR Contract N00014-81-K-0004. For further information see HPP-80-18, HPP-80-24, HPP-82-27, and HPP 83-26.

2.3. The AGE Project

The primary objective of the AGE (Attempt to GEneralize) Project was to build a software laboratory designed to speed up the process of constructing expert systems. This involved two major tasks:

- Tool Building--to isolate the inference, control, and representation techniques used in other expert systems and extract the domain independent portions for use in new domains.
- User Interface Building--to build an intelligent front-end to guide the user in constructing expert systems with the domain independent tools.

2.3.1. System Organization

The current AGE system provides the user with a set of pre-programmed modules called *components*. Using different combinations of components, the user can build a variety of programs that display different problem-solving behavior. AGE also provide user interface modules that help the user in constructing and specifying the details of the components.

A component is a collection of functions and variables that support conceptual entities in

program form. For example, a production rule, as a component, consists of a rule interpreter and various strategies for rule selection and execution. The components in AGE have been selected and modularized to be useable in combinations. For novice users, AGE currently provides the user two predefined configurations of components called *frameworks*. One, called the *Blackboard framework*, is for building programs based on the Blackboard Model with a globally accessible data structure called a *blackboard* and independent sources of knowledge which cooperate to form hypotheses. The other framwork, called the *Backchain framework*, is for building programs that use backward-chaind production rules as the primary mechanism of generating inferences.

To support the user in the selection, specification, and use of compoents, AGE is organized around five major subsystems that interact in various ways. These subsystems are: Browse, Design, Acquisition, Interpreter, and Explanation. A system executive allows the user to access the subsystems through the use of menu selection.

The Browse and Design subsystems help familiarize the user with AGE and guide him in the construction of expert systems through the use of predefined framworks. The Acquisition subsystem is a collection of interface modules that help the user specify the various components of the framework. The Interpreter subsystem is designed for executing, testing, and refining the user program. The Explanation subsystem answers questions about the execution of the user's program.

In addition, AGE provides an interface to the the Units System (developed within the MOLGEN Project at the HPP), an object-oriented, frame-based knowledge representation tool. This gives the user additional flexibility in representing knowledge as well as providing an alternative hypothesis structure for blackboard-based systems.

2.3.2. AGE Status

AGE is fully implemented as a first-generation system. It has been used in medical domains like the PUFF (Pulmonary Function) system and signal understanding domains like HASP/SIAP (a system for locating ships at sea based upon sonar signals). It is especially useful for comparing problem-solving problems on a single problem; PUFF was tested using the built-in forward-chaining and backward-chaining frameworks and a custom assembled model-based framework. Current work is proceeding on an AGE-II systems which is totally useable by a domain expert with no applied artificial intelligence training.

2.4. The MOLGEN Project

The MOLGEN project is a joint effort among computer scientists and molecular biologists to explore applications of artificial intelligence to problems in the rapidly expanding domain of molecular genetics. A central theme of this period of the research was the problem of designing laboratory experiments: producing an ordered list of steps, that when implemented in a biological laboratory, would satisfy a given goal in analysis or synthesis.

During the first year of this period, initial work was completed on two quite different planning systems. The first was based upon a study on how human scientists design experiments. This resulted in a theory known as *skeletal plans*, the idea that almost all experimental designs resulted from the instantiation of an abstract design with specific laboratory steps suitable to the exact strategic and environmental conditions of the experiment. The skeletal plans themselves were taken as part of the scientist's personal knowledge base, not to be recreated for each new experiment; for example, there is strong evidence that a single abstract design for cloning experiments forms the basis for the great majority of all such experiments performed. The design system operates by locating a potentially relevant skeletal plan from its knowledge base and then refining that plan by proceeding hierarchically through a separate knowledge base of laboratory tools and techniques. For more information about this system, see HPP-79-29.

The second planning system was based more upon computer science theoretic grounds, making maximal use of the interactions among potential plan steps and the contraints they impose upon the growing plan. This work led to a design methodology called *constraint*

propagation and a global structure for planning known as *metaplanning*, which separated planning decisions into domain-dependent and domain-independent classes. For more information about this system, see HPP-80-2.

Both of the above-mentioned systems were operational on actual problems in molecular biology, however, considerable work was conducted to expand the scope of the biological knowledge base to allow a wider variety of problems, and more difficult problems to be handled by the systems. The knowledge base increased in size by an order of magnitude during this time period, with particular emphasis on synthetic problems of current interest to our research collaborators.

In addition, new work was begun on extracting the best features of the two planning systems and producing a new, second-generation design system. This system, called SPEX, was finished at the end of the contract period, and was tested successfully on a variety of problems and domains. For more information, see HPP-82-22.

MOLGEN work was co-funded by NSF grant ECS-80-16247.

2.5. The CRYSALIS Project

The task of interpreting three-dimensional signal information is a difficult one for knowledge-based systems because of the combinatoric explosion that results from the vast amount of knowledge that must be applied to solve all real-world problems of this type. Research into this problem occurred within the task domain of X-ray protein crystallography under the CRYSALIS Project. Here the goal is to interpret a three-dimensional image of the electron density cloud surrounding a molecule in order to determine the precise location of the individual atoms which form the molecule.

2.5.1. Problem-Solving Architecture

A major achievment of CRYSALIS came in solving the problem of how the limited resources of a signal interpretation sytem should be allocated when many plausible choices exist. This is known as the *focus-of-attention* problem. A solution was developed using the expert's strategic knowledge to guide the system's problem-solving activities. These control heuristics are represented (as is all other knowledge in CRYSALIS) as production rules and are organized into a hierarchical production system.. Control in this architecture proceeds from the top down, through many levels of control down to object-level heuristics at the bottom of the hierarchy. Each level is a complete production system that examines the current situation and invokes one or more sets of rules at the next lower level. As control moves from general to very specific strategies, from a broad to a very narrow view of the situation, focus of attention is achieved in a very clear and efficient manner.

The particular architecture chosen was the blackboard architecture first demonstrated in the HEARSAY-II speech understanding system. Knowledge in CRYSALIS is partitioned into independent knowledge sources (KSs). These KSs communicate by means of a global database called the *blackboard* which contains a multi-level hypothesis structure and the data. The CRYSALIS system is data-driven in that KSs react only to changes in the data on the blackboard. An important point is that the strategic rules that determine the flow of control from one level of the hierarchy to another are themselves KSs in the blackboard system (known as *control knowledge sources*).

2.5.2. Current Status

At the end of this report period, CRYSALIS was a successful demonstration system. It can solve (i.e. find atom locations) for medium-sized protein in about a day of computing on a dual DEC KI-10 computer system. This is considerably better than traditional numeric systems on far larger computers. The system contains 66 knowledge sources with 602 rules of inference. The system is integrated with several FORTRAN pre-processing programs that skeletonize the orginal electron density map, find its critical points, and create the data representations used as input by CRYSALIS.

CRYSALIS work was co-funded by NSF Grant MCS79-33666. Further information may be

found in HPP-79-16 and HPP-83-19.

2.6. The Knowledge-Based VLSI Project--KB-VLSI

The KB-VLSI project is directed toward the development of AI techniques and applications to computer aided design tools for integrated circuit design. The overall goals of the project are:

- To identify and articulate expert knowledge used in integrated circuit design.
- To develop methods for representing and reasoning with this knowledge.
- To develop knowledge-based expert systems for assisting in the integrated circuit design, test, and debug cycle.

In a broader sense, the KB-VLSI project is concerned with derivation of and experimentation with knowledge-based system paradigms appropriate for design-synthesis tasks. The major part of the KB-VLSI activity during the course of this contract concentrated on the development of Palladio, an experimental, but operational, knowledge-based design system.

The KB-VLSI project is a collaborative project involving the HPP, Xerox Palo Alto Research Center, and Fairchild Advanced Research Labs.

2.6.1. AI Issues

The domain of integrated circuit design and testing is particularly rich for development of and experimentation with new AI techniques. The AI research areas of greatest relevance to the project are:

- The "natural language" most often used for specifying the structure of an IC is graphical rather than textual. Few current AI systems exploit the power and flexibility of high-resolution graphics devices. Research and development on intelligent graphics interfaces for entering, editing, storing, and perusing complex knowledge base involves a spectrum of AI concerns ranging from cognitive psychology through formal languages and knowledge representation systems. Such interfaces would be applicable to a variety of AI system, and they are necessary for intelligent CAD systems.
- The specification of an IC includes structural and behavioral descriptions of the circuit. These specifications may be hierarchical with respect to a part-of hierarchy as well as contain descriptions of the circuit at various levels of abstraction. Current representation systems are, at best, only marginally capable of representing effectively such complex specifications. This problem is further complicated by the need to somtimes concurrently consider alternative specifications of a circuit.
- Much of the expert knowledge used in IC design is in the form of tradeoffs rather than constraints. Although there has been work done on designing with constraints, little has been done on using tradeoffs.

2.6.2. Palladio's Model of the Design Process

The creation of behavioral and structural specifications of a circuit usually involves a sequence of transformations from abstract specifications to more detailed implementational description. For example, the design of a combinational logic circuit may involve first transforming a specification of the circuit in terms of boolean equations which relate inputs and outputs into a specification in terms of logic gates and interconnection networks. This may then be transformed into a layout specification expressed in terms of "colored" rectangles.

A useful metaphor for this transformation process is that design is search. The designer searches in a solution space of implementation specifications. Moves in this space are design decisions. Each decision involves considering alternative implementations, testing the alternatives against the contraints and goals imposed by the abstract specifications, and using

tradeoffs to differentiate between "satisficing" alternatives and to resolve conflicts between incompatible constraints and goals. This process is difficult because the solution space is large, the generation of alternative solutions is expensive, information is incomplete, and it is impossible to predict all of the consequences of a decision.

2.6.3. Design Hierarchies

IC designers have, in part, coped with the difficulty of design decisions by exploiting hierarchies in the design process. One powerful technique is to decompose a device into semi-independent subdevices and to focus attention on each individually. For example, a 4-bit register can be considered as four 1-bit registers and interconnections. Another way of partitioning the design process is into *description levels*, abstract models of circuits. Each description level provides languages for describing the behavior and structure of a device which suppress particular details of physical implementations of the device. This reduces the complexity of the elements in a solution space and makes generation and comparison of alternatives less expensive.

Description levels also permit a designer to partition concerns by concentrating on subclasses of design decisions. For example, at an architectural level a designer can work out certain storage and communication decisions before worrying about power considerations. Currently, there are four description levels in Palladio: *Layout*, *Clocked Primitive Switches* (CPS), *Clocked Registers and Logic* (CRL), and *Linked Module Abstraction* (LMA). Collectively, these levels factor the concerns of a digital designer.

The most widely used description level in integrated circuit design is the artwork or layout level. This level describes circuits in terms of "colored rectangles" that can be composed to build up large designs. Associated with each colored rectangle is a set of *composition rules*, called layout design rules. These rules provide a shallow model of composition that is based on a deep model of electrical properties and fabrication tolerances. If designers follow the rules, their designs are guaranteed to have adequate physical spacing on a chip.

The layout level has several properties which are useful for the synthesis of designs. First, primitive terms can be combined to form larger terms and subsystems. Second, there are composition rules which define allowed combination of terms. Third, there is a well characterized set of *bugs* which are avoided when the composition rules are obeyed. At the layout level, these bugs correspond to the function and performance problems caused by incorrect physical spacing.

The other three more abstract description levels have properties analogous to those those of the layout level. The CPS level distinguishes between different uses for logic and is concerned with the digital behavior of a system. Different uses of logic include steering logic, clocking logic, and restoring logic. The composition rules at this level prevent bugs of non-digital behavior caused by charge sharing and invalid switching levels. The CRL level is concerned with the composition of combinational register logic. The composition rules at this level preclude various bugs related to clocking in a two-phase system. The LMA level is concerned with the sequencing of computational events in a digital system. It describes the paths along which data can flow, the sequential and parallel actiation of computations, and the distribution of registers. The composition rules at this level preclude bugs such as starting computations before data are ready, and deadlock bugs that arise from the improper use of shared modules.

2.6.4. Design Knowledge Bases

Much of the design of ICs is done by using parts of existing designs, possibly with modification. This technique exploits the fact that common constructs are used in many circuits, e.g., registers, NAND gates, and I/O pads. In Palladio, knowledge about previously defined circuits is kept in community knowledge bases. These knowledge bases also contain rules about the composition and optimization of circuit components, For example, at the CPS level we have developed a knowledge base that includes a collection of prototype logic gates, a set a rules that define allowed composition of these gates, and a set of optimization rules for reducing various costs of circuits composed of networks of gates.

The use of community knowledge bases in Palladio is supported by the LOOPS system, an

object and data oriented programming system implemented in InterLisp. LOOPS was created, in particular, to support a design environment in which knowledge bases are shared and can be incrementally updated.

2.6.5. Design Evolution

The design of an integrated circuit is an evolutionary process that follows an iterative cycle: create a candidate design, test the design against current requirements, modify the design and/or requirements to create a new candidate design. A design system should have facilities for interactive simulation to provide rapid feedback between proposed changes and their exercise on test cases.

Within Palladio, we have built interactive, rule-based symbolic circuit simulators. These simulators use symbolic reasoning on a hierarchy of behavioral and structural specifications for a circuit in order to predict the outputs of the circuit given a set of inputs. The simulators include a dynamic display capability.

2.6.6. Status of KB-VLSI

The supporting framework for Palladio has been fully built. This includes LOOPS, a high-level, object-oriented graphics package called HILGA, and GLISP (discussed earlier in this report) which provides LOOPS with optimized data and procedure access.

Prototype community knowledge bases for the CPS and LMA levels are completed. Initial knowledge bases for the layout and CRL levels are under development. A rule-based design editor for the CPS level has been implemented.

Rule-based symbolic simulators for the LMA and CPS levels are operational. Palladio is now useful for the design of at least simple, "student-level" integrated circuits.

Further information about the KB-VLSI project may be found in HPP-82-2, HPP-82-5, and HPP-82-11.

2.7. The *Handbook of Artificial Intelligence*

Incorporating the efforts of nearly 200 computer science researchers as writers, editors, and reviewers, the *Handbook of Artificial Intelligence* is an encyclopedic compilation of articles covering the entire field of artificial intelligence. It satisfies the urgent need for AI to "go public," making the full range of its important techniques and concepts available for the first time to the rapidly expanding world of potential users. Its scope, readibility, and organization have made it the standard reference work in AI for both newcomers and experienced members of the research community. It also provides the most comprehensive survey of the field's literature available.

The work consists of approximately 1500 pages in three volumes. Volume I, released in 1981, contains major sections on search, knowledge representation, and understanding natural and spoken language. Volume II, released in 1982, discusses AI programming languages, applications of AI to science, medicine, and education, and automatic programming. Volume III, also released in 1982, contains chapters on cognitive models, deduction, vision, learning, planning, and problem-solving.

All three volumes were published by William Kaufmann, Inc. of Los Altos, California. To date, approximately 90,000 copies have been sold. Royalties from the handbook provide funding for HPP students to attend artificial intelligence conferences. Co-funding for the production of the handbook was provided by the NIH Bureau of Research Programs. Many of the individual chapters are available as HPP Reports.

2.8. Technology Transfer

The HPP has maintained a consistent record of active transfer of its ideas and systems into the academic, governmental, and industrial sectors. For all of the projects described in this report, publications have appeared both in the computer science and the domain-specific

literature. HPP scientists have organized and participated in innumerable symposia, conferences, and workshops.

Many of the systems described in this report have been distributed, at copying and shipping cost, to other sites. In particular, EMCYIN, AGE, MRS, and GLISP have been sent to over 100 locations, including such DOD-related companies as Fairchild, RCA, ESL, Mitre, HP, Honeywell, GTE, Hughes, IBM, Lockheed, NCR, Boeing, and Systems Control.

2.9. HPP Publications

HPP 79-17 STAN-CS-79-749, William Clancey, James Bennett, and Paul Cohen; **"Application-Oriented AI Research: Education,"** in A. Barr, and E. Feigenbaum (eds.), in *Handbook of Artificial Intelligence, Volume I, Los Altos, CA: Wm. Kaufmann, Inc.*, 1981.

HPP 79-18 Larry M. Fagan, Edward H. Shortliffe, and Bruce G. Buchanan; **"Computer-Based Medical Decision Making: From MYCIN to VM,"** in *Automedica, 3, pp. 97-106*, 1980.

HPP 79-19 S. Jerrold Kaplan; **"Cooperative Responses From a Portable Natural Language Data Base Query System,"** Ph.D. dissertation, University of Pennsylvania, July 1979.

HPP 79-20 Edward H. Shortliffe, Bruce G. Buchanan, and Edward Feigenbaum; **"Knowledge Engineering for Infectious Disease Therapy Selection,"** in *Proceedings of the IEEE, Vol. 67, No. 9*, September 1979.

HPP 79-21 STAN-CS-79-754, Anne Gardner, James Davidson, and Terry Winograd; **"Natural Language Understanding,"** in *Handbook of Artificial Intelligence, Volume I, Chapter 4, pp. 223-321, Los Altos, CA: Wm. Kaufmann, Inc.*, 1981.

HPP 79-22 STAN-CS-79-756, James S. Bennett, Bruce G. Buchanan, and Paul R. Cohen; **"Applications-Oriented AI Research: Sciences and Mathematics,"** in A. Barr, and E. Feigenbaum (eds.), *Handbook of Artificial Intelligence, Volume II, Los Altos, CA: Wm. Kaufmann, Inc.*, 1982.

HPP 79-23 STAN-CS-79-757, Victor Ciesielski, James S. Bennett, and Paul R. Cohen; **"Applications-Oriented AI Research: Medicine,"** in A. Barr, and E. Feigenbaum (eds.), *Handbook of Artificial Intelligence, Volume II, Los Altos, CA: Wm. Kaufmann, Inc.*, 1982.

HPP 79-24 STAN-CS-79-758, Robert Elschlager, and Jorge Phillips; **"Automatic Programming,"** in A. Barr, and E. Feigenbaum (eds.), *Handbook of Artificial Intelligence, Volume II, Los Altos, CA: Wm. Kaufmann, Inc.*, 1982.

HPP 79-25 STAN-CS-79-759, Alain Bonnet; **"Schema-Shift Strategies to Understanding Structured Texts in Natural Language,"** submitted to *American Journal of Computational Linguistics*, August 1979.

HPP 79-27 Ramez El-Masri, and Gio Wiederhold; **"Data Model Integration Using the Structural Model,"** in *ACM-SIGMOD 1979, pp. 191-202*, May 30-June 1, 1979.

HPP 79-28 Bruce G. Buchanan; **"Steps Toward Mechanizing Discovery,"** presented at the *Pittsburgh University Conference on the Logic of Diagnosis*, October 1978.

HPP 79-29 STAN-CS-79-771, Peter E. Friedland; **"Knowledge-Based Experiment Design in Molecular Genetics,"** Ph.D. dissertation, Stanford University, October 1979.

HPP 79-30 Jonathan King; **"Exploring the Use of Domain Knowledge for Query Processing,"** December 1979, in *ACM SIGMOD Conference,* May 14-16, 1980.

HPP 79-31 L.M. Fagan, J.C. Kunz, E.A. Feigenbaum, and J.J. Osborn; **"Representation of Dynamic Clinical Knowledge: Measurement Interpretation in the Intensive Care Unit,"** in *Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Tokyo, Japan,* August 1979.

HPP 79-32 L.M. Fagan, J.C. Kunz, and E.A. Feigenbaum; **"A Symbolic Processing Approach to Measurement Interpretation in the Intensive Care Unit,"** in *Proceedings of the Third Annual Symposium on Computer Applications in Medical Care, Silver Spring, Maryland,* October 1979.

HPP 80-1	Avron Barr; **"The Representation Hypothesis,"** in *Proceedings of the CSCSI Conference, Victoria, B.C.*, April 1980, **4 pages.**
HPP 80-2	STAN-CS-80-784, Mark Jeffrey Stefik; **"Planning With Constraints,"** Ph.D. dissertation, Stanford University, January 1980, **239 pages.**
HPP 80-3	Avron Barr, and James Davis; **"Representation of Knowledge,"** in *Handbook of Artificial Intelligence, Vol. I, pp. 143-222, Los Altos, CA: Wm. Kaufmann, Inc.* 1981, **80 pages.**
HPP 80-4	STAN-CS-80-842, Michael Genesereth; **"The Role of Plans in Intelligent Teaching Systems,"** in Derek Sleeman (ed.), *Intelligent Teaching Systems, Academic Press,* April 1980, **19 pages.**
HPP 80-5	E.A. Feigenbaum, B.G. Buchanan, L.H. Kedes, and D. Brutlag; **"A Computer Science Application to Molecular Biology,"** submitted to *National Science Foundation,* April 1980, **114 pages.**
HPP 80-6	Sharon W. Bennett, and A. Carlisle Scott; **"Computer-Assisted Customized Antimicrobial Dosages,"** in *American Journal of Hospital Pharmacy, Vol. 37, pp. 523-530,* April 1980, **8 pages.**
HPP 80-7	J.S. Bennett, and D. Goldman; **"CLOT: A Knowledge-Based Consultant for Diagnosis of Bleeding Disorders,"** April 1980, **8 pages.**
HPP 80-8	David E. Smith; **"CORLL: A Demand Paging System for Units,"** April 1980, **11 pages.**
HPP 80-9	Russell Greiner; **"RLL-1: A Representation Language Language,"** in *AAAI-1, pp. 165-169,* 1980, **44 pages.**
HPP 80-10	W.J. Clancey, E.H. Shortliffe, and B. G. Buchanan; **"Intelligent Computer-Aided Instruction for Medical Diagnosis,"** in *Third Computer Application in Medical Care, pp. 175-183,* October 1979, **10 pages.**
HPP 80-11	W. van Melle, A.C. Scott, J. S. Bennett, and M. Peairs; **"The EMYCIN Manual,"** October 1981. **(Superseded by HPP 81-16)**
HPP 80-12	Mark Stefik; **"Planning With Constraints - MOLGEN: Part 1,"** in *Artificial Intelligence Journal,* June 1980, 33 pages.
HPP 80-13	Mark Stefik; **"Planning and Meta-Planning - MOLGEN: Part 2,"** in *Artificial Intelligence Journal, 16:2,* June 1980. Also in Webber & Nilsson's book *Readings in Artificial Intelligence,* **34 pages.**

HPP 80-14 E.H. Shortliffe; "The Computer as Clinical Consultant," in *Archives of Internal Medicine, Vol. 140, pp. 313-314,* March 1980, 2 pages.

HPP 80-15 A. Carlisle Scott, M. Bischoff, and E.H. Shortliffe; **"Oncology Protocol Management Using the Oncocin System: A Preliminary Report,"** July 1980, 25 pages.

HPP 80-17 STAN-CS-80-814, Janice Aikins; **"Prototypes and Production Rules: A Knowledge Representation for Computer Consultations,"** Ph.D. dissertation, Stanford University, August 1980, 208 pages.

HPP 80-18 Michael Genesereth, and Douglas Lenat; **"Meta-Description and Modifiability in a Knowledge Representation System,"** submitted to *Artificial Intelligence,* August 1980.

HPP 80-19 Michael Genesereth; **"Why,"** August 1980.

HPP 80-20 Michael Genesereth; **"Metaphors and Models,"** in *Proceedings of the First National Conference on Artificial Intelligence, Stanford, CA.,* August 1980. Also in *AAAI-80,* 9 pages.

HPP 80-21 STAN-CS-80-812, Edward A. Feigenbaum; **"Knowledge Engineering: The Applied Side of Artificial Intelligence,"** September, 1980, 27 pages.

HPP 80-22 William van Melle; **"A Domain-Independent System That Aids in Constructing Knowledge-Based Consultation Programs,"** Ph.D. dissertation, Stanford University, June 1980.

HPP 80-23 Russell Greiner, and Douglas B. Lenat; **"Details of RLL-1,"** October 1980, 66 pages.

HPP 80-24 Michael R. Genesereth, Russel Greiner, and David Smith; **"MRS Dictionary,"** current version, October 1981, 49 pages. Growth, 12, pp. 148-165,] 1984, 57 pages.

HPP 80-26 Douglas Lenat; **"Toward a Theory of Heuristics,"** to appear in *Proceedings of the Bern. Conferences on Heuristics,* 1981. Also submitted to *Artificial Intelligence Journal,* November 1980. **(Revised to HPP 81-22)**

HPP 80-27 Douglas Lenat; **"The Plausible Mutation of DNA,"** November 1980, 34 pages.

HPP 80-28 Reid G. Smith, and Peter Friedland; **"Unit Package User's Guide,"** December 1980, 105 pages.

HPP 80-29 H. Penny Nii; **"An Introduction to Knowledge Engineering, Blackboard Model**

and AGE," March 1980, **45 pages.**

HPP 80-30 E.H. Shortliffe; **"Consultation Systems for Physicians,"** in *Proceedings of the CSCSI/SCEIO Conference (Canadian Society for the Computer Simulation of Intelligence), University of Victoria, British Columbia, pp. 1-11,* May 14-16, 1980, **12 pages.**

HPP 80-31 L.M. Fagan; **"VM: Representing Time-Dependent Relations in a Clinical Setting,"** Ph.D. dissertation, Stanford University, 1980.

HPP 80-32 E.H. Shortliffe; **"ONCOCIN Annual Report (1979-80): (a) Summary Progress Report. (b) Full Progress Report With Appendices,"** April 1980.

HPP 80-33 E.H. Shortliffe; **"Medical Computing: Another Basic Science?"** In *Proceedings of the 4th Symposium on Computer Applications in Medical Care, Washington, D.C.,* November 1980, **4 pages.**

HPP 81-1 STAN-CS-81-837, Bruce G. Buchanan; **"Research on Expert Systems,"** in J. Hayes, D. Michie, and Y-H Pao (eds.), *Machine Intelligence 10, pp. 269-299, Chichester: Ellis Horwood Ltd.,* 1982, **39 pages.**

HPP 81-2 STAN-CS-82-908, William J. Clancey, and Reed Letsinger; **"NEOMYCIN: Reconfiguring a Rule-Based Expert System for Application to Teaching,"** February 1981, in *Proceedings of the 7th International Joint Conference on A.I., Vancouver, British Columbia,* August 1981, **14 pages.**

HPP 81-3 Jan Clayton, Peter Friedland, Laurence Kedes, and Doug Brutlag; **"SEQ- Sequence Analysis System,"** April 1981, **20 pages.**

HPP 81-4 David R. Barstow, and Bruce G. Buchanan; **"Maxims for Knowledge Engineering,"** May 1981, in R. Hayes-Roth, D. Waterman, and D. Lenat (eds.), *Building Expert Systems, Addison-Wesley,* 1983, **7 pages.**

HPP 81-5 Motoi Suwa, A. Carlisle Scott, and Edward H. Shortliffe; **"An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System,"** June 1981, in *AI Magazine 3 (4), pp. 16-21,* 1982, **23 pages.**

HPP 81-6 Michael R. Genesereth, and David E. Smith; **"Meta-Level Architecture,"** current version, December 1982. **(Replaced by HPP 84-1)**

HPP 81-7 Edward H. Shortliffe, A. Carlisle Scott, Miriam B. Bischoff, A. Bruce Campbell, William van Melle, and Charlotte D. Jacobs; **"ONCOCIN: An Expert System for Oncology Protocol Management,"** in *Proceedings of the 7th IJCAI-81, pp. 876-881,* August 1981, **9 pages.**

HPP 81-8 James S. Bennett, and Clifford R. Hollander; **"DART: An Expert System for Computer Fault Diagnosis,"** in *Proceedings of IJCAI-81, pp. 843-845,* August 1981, **8 pages.**

HPP 81-9 Edward H. Shortliffe; **"Evaluating Expert Systems,"** July 1981, in R. Hayes-Roth, D.A. Waterman, and D.B. Lenat (eds.), *Building Expert Systems, Chapter 8, Addison Wesley, Boston, MA.,* 1983, **22 pages.**

HPP 81-10 Randy L. Teach, and Edward H. Shortliffe; **"An Analysis of Physician Attitudes Regarding Computer-Based Clinical Consultation Systems,"** in *Computer Biomed. Res. 14, pp. 542-558,* March 1981, **17 pages.**

HPP 81-11 Michael R. Genesereth, Milton R. Grinberg, and Jay S. Lark; **"SUBTLE Manual." (Revised to HPP 84-37)**

HPP 81-12 Neil A. B. Gray; **"Applications of Artificial Intelligence for Organic Chemistry: Analysis of C-13 Spectra,"** October 1981, **112 pages.**

HPP 81-13	Thomas G. Dietterich, and Richard S. Michalski; **"Inductive Learning of Structural Description: Evaluation Criteria and Comparative Review of Selected Methods,"** in *Artificial Intelligence 16, pp. 257-294,* 1981, **37 pages.**
HPP 81-14	STAN-CS-81-879, Larry M. Masinter; **"Interlisp VAX: A Report,"** December 1981, **13 pages.**
HPP 81-15	J.J. Finger; **"Planning in a Dangerous World,"** February 1982, **16 pages.**
HPP 81-16	STAN-CS-81-885, William van Melle, A. Carlisle Scott, James S. Bennett, and Mark A. S. Peairs; **"The EMYCIN Manual," 143 pages. (Replaces HPP 80-11)**
HPP 81-17	STAN-CS-81-896, William J. Clancey; **"The Epistemology of a Rule-Based Expert Systems: A Framework of Explanation,"** December 1981, in *Artificial Intelligence 20, pp. 215-251,* 1983, **36 pages.**
HPP 81-18	**STAN-CS-81-894,** William J. Clancey; **"Methodology for Building an Intelligent Tutoring System,"** December 1981, in Walter Kintsch, James R. Miller, and Peter G. Polson (eds.), *Method and Tactics in Cognitive Science, pp. 51-83, Hillsdale, N.J.,: Lawrence Erlbaum Associates,* 1984, **32 pages.**
HPP 81-19	**STAN-CS-81-891,** T.G. Dietterich, and B.G. Buchanan; **"The Role of the Critic in Learning Systems,"** December 1981, **26 pages.**
HPP 81-20	Michael R. Genesereth; **"The Use of Hierarchical Design Models in the Automated Diagnosis of Computer Systems,"** December 1981, **28 pages.**
HPP 81-21	Douglas Lenat, Randal Davis, Jon Doyle, Michael Genesereth, Ira Goldstein, and Howard Schrobe; **"Meta-Cognition: Reasoning About Knowledge,"** December 1981, **23 pages.**
HPP 81-22	Douglas Lenat; **"The Nature of Heuristics,"** December 1981, in *Artificial Intelligence 19, pp. 189-249,* 1982, **61 pages.**
HPP 81-23	Nelleke Aiello, Conrad Bock, H. Penny Nii, and Wm. C. White; **"Joy of AGE-ing: An Introduction to the Age-1 System,"** October 1981, **106 pages.**
HPP 81-24	Nelleke Aiello, Conrad Bock, H. Penny Nii, and Wm. C. White; **"AGE Reference Manual,"** October 1981, **125 pages.**
HPP 81-25	Nelleke Aiello, and H. Penny Nii; **"AGEPUFF: A Simple Event-Drive Program,"** October 1981, **64 pages.**

HPP 81-26 Nelleke Aiello, and H. Penny Nii; **"BOWL: A Beginner's Program Using AGE,"** October 1981, **50 pages.**

HPP 81-27 W. van Melle, E.H. Shortliffe, and B.G. Buchanan; **"EMYCIN: A Domain-Independent System that Aids in Constructing Knowledge-Based Consultation Programs,"** in *Pergamon-Infotech State of the Art Report on Machine Intelligence,* August 1981, **24 pages.**

HPP 81-28 E.H. Shortliffe; **"ONCOCIN Annual Report (1980-81): (a) Summary Progress Report. (b) Full Progress Report with Appendices,"** April 1981.

HPP 81-29 E.H. Shortliffe, A.B. Campbell, P. Chang, J. Cho, D. Hickam, and R. Teach; **"Proposal for the Evaluation of the ONCOCIN System,"** June 1981.

HPP 81-30 E.H. Shortliffe, and B.G. Buchanan; **"MYCIN Project Annual Report,"** May 1981.

HPP 81-31 E.H. Shortliffe; **"Expert Systems Research: Adapting Technical Knowledge for Computer-Based Consultation Systems,"** in *Proceedings of the Sixth Symposium on Legal Data Processing, Sponsored by the Council of Europe, Thessaloniki, Greece,* July 1981, **24 pages.**

HPP 81-32 E.H. Shortliffe; **"ONCOCIN: An Aid for the Outpatient Management of Cancer Patients,"** in *Proceedings of the Society for Computer Medicine, pp 57-58, Washington, D.C.,* October 1981, **3 pages.**

HPP 82-1 STAN-CS-82-895, Gordon S. Novak, Jr.; **"GLISP User's Manual,"** January 1982, **39 pages.**

HPP 82-2 Mark Stefik, Daniel Bobrow, Alan Bell, Harold Brown, Lynn Conway, and Christopher Tong; **"The Partitioning of Concerns in Digital System Design,"** January 1982, **12 pages.**

HPP 82-3 Edward H. Shortliffe, and Larry M. Fagan; **"Expert Systems Research: Modelling the Medical Decision Making Process,"** February 1982, in *An Integrated Approach to Monitoring, pp. 183-200,* 1983, **24 pages.**

HPP 82-4 Jay S. Lark; **"OMEN User's Manual,"** February 1982, **15 pages.**

HPP 82-5 Harold Brown, and Mark Stefik; **"PALLADIO: An Expert Assistant for Integrated Circuit Design,"** April 1982, **7 pages. (Superseded by HPP 83-31)**

HPP 82-6 H. Penny Nii, E.A. Feigenbaum, J. Anton, and A.J. Rockmore; **"Signal-to-Signal Transformation: HASP/SIAP Case Study,"** in *AI Magazine, Spring 1982, pp. 23-35,* April 1982, **12 pages.**

HPP 82-7 **STAN-CS-82-909,** Bob London, and William J. Clancey; **"Plan Recognition Strategies in Student Modeling: Prediction and Description,"** May 1982, **12 pages.**

HPP 82-8 **STAN-CS-82-910,** William J. Clancey, and Bruce G. Buchanan; **"Exploration of Teaching and Problem-Solving Strategies, 1970-1980,"** May 1982, **18 pages.**

HPP 82-9 David E. Smith, and Michael R. Genesereth; **"Ordering Conjunctive Queries,"** to appear in *Artificial Intelligence,* October 5, 1984, **46 pages.**

HPP 82-10 Thomas G. Dietterich, Bob London, Kenneth Clarkson, and Geoff Dromey; **"Learning and Inductive Inference,"** in *Handbook of Artificial Intelligence, Vol. 3,* May 1982, **187 pages.**

HPP 82-11 Christopher Tong; **"Framework for Design,"** June 1982, **8 pages.**

HPP 82-12 J.J. Finger; **"Sensory Planning,"** April 1982, **9 pages.**

HPP 82-13 **STAN-CS-82-931,** J.Aikins, J. Kunz, E.H. Shortliffe, and R.J. Fallat **"PUFF: An Expert System for Interpretation of Pulmonary Function Data,"** July 1982, in *Computer Biomed. Res., 16, pp. 199-208,* 1983, **22 pages.**

HPP 82-14 B.G.Buchanan, and R. Duda; **"Principles of Rule-Based Expert Systems,"** in M. Yovits (ed.), in *Advances in Computers, Vol.22, pp. 163-216, New York: Academic Press,* August 1982, **57 pages.**

HPP 82-15 E.Schoen; **"The Dolphin Site Manager's Guide,"** June 1982, **24 pages.**

HPP 82-16 STAN-CS-82-934, A.Barr, P.Cohen, and L. Fagan; **"Understanding Spoken Language,"** in *Handbook of Artificial Intelligence, Vol. I, Chap. 5,* July 1982, **50 pages.**

HPP 82-17 STAN-CS-82-935, S.Tappel, S. Westfold, and A. Barr; **"Programming Languages for AI Research,"** in *Handbook of Artificial Intelligence, Vol. II, Chap. 6,* July 1982, **90 pages.**

HPP 82-18 STAN-CS-82-936, P. Cohen; **"Models of Cognition,"** in *Handbook of Artificial Intelligence, Vol. III, Chap. 11,* July 1982, **87 pages.**

HPP 82-19 STAN-CS-82-937, M.Ballantyne, W.W. Bledsoe, J. Doyle, R.C.Moore, R. Pattis, and S.Rosenschein; **"Automatic Deduction,"** in *Handbook of Artificial Intelligence, Vol. III, Chap. 12,* July 1982, **39 pages.**

HPP 82-20 STAN-CS-82-938, T. Kanade; **"Vision,"** in *Handbook of Artificial Intelligence, Vol. III, Chap. 13,* July 1982, **219 pages.**

HPP 82-21 STAN-CS-82-939, P. Cohen; **"Planning and Problem Solving,"** in *Handbook of Artificial Intelligence, Vol. III, Chap. 15,* July 1982, **59 pages.**

HPP 82-22 Y. Iwasaki; **"SPEX: Skeletal Planner of Experiments,"** September 1982, **60 pages.**

3. VLSI Theory/Silicon Compilation Project

As described in the introduction to this report, the VLSI Theory/Silicon Compilation work was performed outside of the HPP, under the direction of Professor Jeffrey Ullman. A separate references list is included at the end of this section and referred to within the text.

3.1. Regular Expression Compilation

The basic idea is to translate regular expressions into control structures, such as PLA's. The regular expression language is an easy to describe collections of processes that independently look for patterns on a shared input. In a sense, regular expressions are an alternative to conventional finite state machine description languages. They are awkward for some things, but they are also elegant and natural for other purposes. For example, they are a natural way to describe communication protocols or processor control units.

The project has considered and tried out a number of ways to translate these expressions to silicon, but the central theme is that regular expressions are translated into nondeterministic automata, and the nondeterministic automata have their states encoded in a way that allows the automation to be implemented by a convetional form of logic, such as PLA's. (See[1] for definitiions and simple introductions to the necessary concepts, such as regular expressions and non-deterministic automata.)

The first attempt to build a regular expression compiler was based on the idea that a small circuit could be laid out corresponding to each operand, and these circuits could be wired together in systematic ways to reflect the operators of the expression. The ideas were described in [2], but the resulting layouts were found on average to be considerably worse than the other methods we tried later.

Our next approach was to select small subexpressions that could be implemented by PLA's in a simple fashion, and wiring the PLA's together in ways that reflect the operators of the expression, as before. In addition, we used a heuristic search for the "best" subexpressions to choose. These improvements were described in [3,4].

At this point, we realized that coding the states of the nondeterministic automata that correspond to each of the selected subexpressions is a critical problem, and we tried a number of different ways to find efficient codes. [5] discusses an early attempt and also discusses the (deterministic) "state" feature, that enables the regular expression language to include conventional finite state machine languages for controller specification. That paper also described a brief flirtation with the Weinberger array oriented approach of Steve Johnson (Bell Labs). We found that the sizes of circuits obtained by generating Johnson's lgen language from regular expressions was comparable to what we obtained by our own PLA method.

Our best coding method to date is described in [6]. With this approach, we generate PLA's that are superior to hand designs for many of the benchmark problems that we accumulated during the course of the project.

3.2. Routing

We developed a provably optimal and highly efficient algorithm for river-routing in a rectangular channel[7]. These ideas have been extended to wiring rules more general than rectangular wiring, e.g., rules permitting 45-degree wires, and more general configurations, such as the "bristle blocks" problem, where the sides of the channel consists of rigid modules able to slide horizontally relative to one another. These extensions are covered in [8-10].

3.3. PLA Folding

A more general PLA folder, which considers the possibility that input wires need to come in truecomplement pairs, was implemented[11]. Like most PLA folders, it uses a greedy heuristic, finding legal folds and making them. A graph-theoretic representation makes the test for legality of a fold relatively efficient.

3.4. Plane Embeddings

Given a circuit described by nodes, e.g., logic gates, and wires connecting them, we would like to lay out the circuit in minimum possible area. Paper [12] shows how to lay out a class of such circuits in area proportional to the number of nodes; the class is larger than that handled by previously known layout algorithms.

3.5. VLSI-Oriented Algorithms

Paper [13] considers algorithms for doing a number of important operations with a special-purpose chip; these operations are graph-theoretic, such as connected components finding (that problem is equivalent to "circuit extraction" from a CIF layout). There are a number of powerful algorithms ideas, such as "funnel pipelining," where the graph is progressibely transformed by combining two nodes into one, so after log n stages, an n-node graph becomes trivial. At successive stages, the time to perform the transformation doubles, so the total work at each stage is the same, and the stages can be implemented by a pipeline on the chip, where all stages operate in parallel.

3.6. References

[1] J.D. Ullman, *Computational Aspects of VLSI*, Computer Science Press, 1984.

[2] R.W. Floyd and J. D. Ullman, "The compilation of regular expressions into integrated circuits,"*JACM*29:2, pp.603-622, 1982.

[3] H. W. Trickey, "Good layouts for pattern recognizers," *IEEE Trans. Computers* C-31:6,pp.514-520, 1981.

[4] H. W. Trickey and J. D. Ullman, "A regular expressions compiler," *IEEE COMPCON*, pp.345-348, 1982.

[5] J. D. Ullman, "Combining state machines with regular expressions for autmatic synthesis of VLSI circuits," STAN-CS-927, 1982.

[6] A. R. Karlin, H. W. Trickey, and J. D. Ullman, "Experience with a regular expression compiler," *IEEE ICCD*, pp.656-665, 1983

[7] D. Dolev, K. Karplus, A. Siegel, A. Strong, and J. Ullman, "Optimal wiring between rectangles," *Proc. 13th Annl. ACM Sym. on Theory of Computing*, pp.312-317, 1981

[8] A. R. Siegel, "Fast optimal placement for river routing," unpublished memorandum, Dept. of CS, Stanford Univ., 1983.

[9] A. R. Siegel, and D. Dolev, "The separation required for general single-layer wiring barriers,"*VLSI Systems and Computations* (Kung, Sproul,and Steele, eds.), Computer Science Press, 1981

[10] A. R. Siegel, "River routing for VLSI: the theory and methodology," Ph.D. thesis, Dept. of CS, Stanford Univ., 1983.

[11] A. R. Siegel, program documentation, unpublished, Stanford Univ.

[12] D. Dolev, F. T. Leighton, and H. W. Trickey, "Planar embedding of planar graphs," unpublished memorandum, Dept. of CS, Stanford Univ.

[13] P. H. Hochschild, E. W. Mayr, and A. R. Siegel, "Techniques for solving

graph problems in parallel environments," *Proc. 24th IEEE Symp. on Foundations of CS*, pp.351-359, 1983.